Analysis of TTL-based Cache Networks

Nicaise Choungmo Fofack, Philippe Nain, Giovanni Neglia, Don Towsley

Analysis of TTL-based Cache Networks

Nicaise Choungmo Fofack, Philippe Nain, Giovanni Neglia, Don Towsley

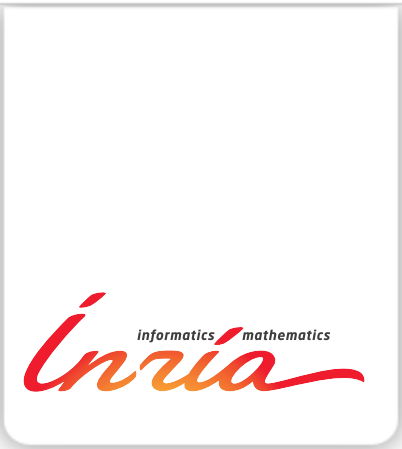
RESEARCH REPORT

N° 7883

March 2012

Project-Team MAESTRO

ISSN 0249-6399 ISRN INRIA/RR--7883--FR+ENG



Analysis of TTL-based Cache Networks

Nicaise Choungmo Fofack*, Philippe Nain*, Giovanni Neglia*,
Don Towsley†

Project-Team MAESTRO

Research Report n° 7883 — March 2012 — 24 pages

Abstract: This paper provides building blocks for the performance evaluation of Content Centric-like Networks (CCNs). In CCNs if a cache receives a request for a content it does not store (*miss*), it forwards the request to a higher-level cache, if any, or to the server. When located, the document is routed on the reverse-path and a copy is placed in each cache along the path. In this paper we consider a cache replacement policy based on Time-to-Lives (TTLs) like in a DNS network. A local TTL is set when the content is first stored at the cache and is renewed every time the cache can satisfy a request for this content (at each hit). The content is removed when the TTL expires. Under the assumption that requests follow a renewal process and the TTLs are exponential random variables, we determine exact formulas for the performance metrics of interest (average cache occupancy, hit and miss probabilities/rates) for some specific architectures (a linear network and a tree network with one root node and N leaf nodes). For more general topologies and general TTL distributions, we propose an approximate solution. Numerical results show the approximations to be accurate, with relative errors smaller than 10^{-3} and 10^{-2} respectively for exponentially distributed and constant TTLs.

Key-words: Cache architecture, content-centric network, timer, Markov model, renewal theory.

* Email: {nicaise.choungmo_fofack,philippe.nain,giovanni.neglia}@inria.fr. Postal address: INRIA, B.P. 93, 06902 Sophia Antipolis, Cedex, France. This author was supported in part by Orange Labs under Grant "CRE CCN" number 5376.

† Email: towsley@cs.umass.edu. Postal address: Department of Computer Science, University of Massachusetts, Amherst, MA 01002, USA. This author was supported in part by Inria and by NSF under grant CNS-1040781.

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Analyse de réseaux de caches TTL

Résumé : Cet article développe des briques de base pour l'évaluation des performances de réseaux orientés contenus. Dans ces réseaux lorsqu'un nœud ou cache reçoit une requête pour un contenu qu'il ne possède pas il la transmet à un ou plusieurs caches de niveau supérieur. Une fois le contenu localisé il est envoyé et stocké à tous les caches qui ont reçu la requête ainsi qu'à l'utilisateur. Dans cet article nous nous intéressons à une politique de gestion des caches qui utilise des temporisateurs (TTL pour *Time-to-Live*). A chaque arrivée d'un contenu dans un cache un temporisateur est déclenché. Chaque nouvelle requête pour ce contenu régénére le temporisateur. Dès qu'un temporisateur expire le contenu correspondant est effacé du cache. Nous calculons de manière exacte différentes mesures de performance (occupation moyenne des caches, probabilité et taux de succès) pour des architectures particulières (réseau linéaire, réseau arborescent composé d'une racine et de N feuilles) dans le cas où les requêtes succcessives aux feuilles forment des processus de renouvellement et où les temporisateurs sont exponentiellement distribués. Des approximations très précises (erreurs relatives de l'ordre de 10^{-2}) sont proposées pour des architectures plus générales et/ou des distributions arbitraires des TTL.

Mots-clés : Architecture de caches, réseau orienté contenus, temporisateur, modèle de Markov, théorie du renouvellement.

1 Introduction

Caches are widely used in networks and distributed systems for improving performance. They are integral components of the Web [4], DNS [15], and Content Distribution Networks (CDNs) [23]. More recently there has been a growing emphasis on *content networks* where content is addressable and host-to-content interaction is the common case [22]. Many of these systems give rise to hierarchical (tree) cache topologies and to even more general irregular topologies.

The design, configuration, and analysis of these cache systems pose significant challenges. An abundant literature exists on the performance (e.g. hit probability, search cost) of a single cache running the Least Recently Used (LRU) replacement policy[1] or, its companion, the Move-to-Front (MTF) policy (see [20, 16, 9, 2, 1, 8, 6, 10, 11] for iid requests and [5, 13, 12] for correlated requests). With few exceptions, exact models of even single caches are computationally intractable, resulting in the reliance on approximations [6, 11]. Networks of caches are significantly more difficult to analyze and no exact solution has been obtained so far for even the simple configuration of two LRU caches in series. A few approximations have been proposed, instead, for a simple two-level LRU cache network [4] and a general LRU network [21]; however their accuracies can be poor (relative error up to 16% reported in [21]).

In this paper we focus on a class of caches, referred to as Time-To-Live (TTL) caches. When an uncached data is brought back into the cache due to a cache miss, a TTL is set. All requests to that data before the expiration of the TTL are successful (cache hit); the first request for that data to arrive after the TTL expiration will yield a cache miss. TTL-based caches are known to exhibit high hit rates and good scaling properties and are used in DNS for that reason.

We develop a set of building blocks for the performance evaluation of hierarchical TTL cache networks where TTLs are set with every request. These building blocks allow one to model exogenous requests to different nodes as independent renewal processes and to allow for TTL durations to be described by an arbitrary distribution so long as they are independent of each other.

The building blocks consist of

- a renewal theoretic model of a *single content* TTL cache when fed by a renewal request stream,
- a renewal process approximation of the superposition of independent renewal processes.

The first block forms the basis for calculating cache metrics such as miss and hit probabilities and rates while the second block is used to represent the superposition of exogenous requests and those resulting from a miss at an upstream cache as a renewal process.

We apply these blocks primarily to the case that TTL durations are either constant or exponentially distributed. Furthermore, we focus primarily on linear TTL neworks, two level TTL tree networks and combinations of the two. In some cases our results are exact but when they are not, the relative errors are extremely small ($< 10^{-3}$ in the case of exponentially distributed TTLs and $< 10^{-2}$ in the case of constant TTLs). Thus our approach is extremely promising and we believe capable of accurately modeling a richer class of network topologies. Last, although the approach applies to single content caches, we demonstrate how it can be used to optimize a multi-content cache network.

In the literature the paper closer to our approach is [14], where the authors consider a *single* TTL-based cache fed by iid requests to a single data. They obtain the hit rate for a constant TTL

[1] The replacement policy is the rule used to select the data to eject from the cache. Other popular policies are the Most-Recently-Used (MRU) and Random Remplacement (RR); MRU is more effective than LRU for cyclic access patterns and RR is used in RISC architectures due to its simplicity.

via the solution of a renewal-like equation. Despite the increasing interest in CCNs, previous work has mainly focused on global architecture design. [3] is probably the first attempt to model data transfer in CCNs. The authors develop approximations to calculate the stationary throughput in a network of LRU caches taking into account the interplay between receiver driven transport and per-chunk caching.

The paper is organized as follows. We introduce notation, the model assumptions, and a key result from [19] regarding how to compute the marginal interarrival distribution for a superposition of arrival streams modeled as renewal processes in Section 2. Section 3 contains our renewal theoretic model along with its application to a number of networks where it leads to exact results. Section 4 describes our approach to modeling the combined exogenous/miss request stream as a renewal process and the resulting approximations for a larger class of linear and tree networks. Sections 5 and 6 report on the accuracy of the models and the computational costs of their solutions. Section 7 reports an application of our approach to configure and optimize TTL cache networks. Conclusions are found in Section 8.

2 Definitions and assumptions

Throughout this paper we focus on particular instances of TTL cache tree networks with a *single* data chunk (simply called the data). The case of multiple data will be briefly discussed in Section 7. From now on the words "node" and "cache" will be used interchangeably. Also, a cache will always be a TTL cache unless otherwise specified.

New requests for the data can be generated at any node of the network according to mutually independent renewal processes; these requests are referred to as *exogenous* requests or arrivals. The time instants at which exogenous requests arrive is called the *exogenous request process* or the exogenous arrival process. If upon the arrival of a new request the data is not present in the cache the request is *instantaneously* forwarded to the next level of the tree and the process repeats itself until the data is found. In case the data cannot be found along the path toward the root, the root retrieves it from a server. Once the data is found, either at a cache or at a server, a copy of it is *instantaneously* transmitted to each cache along the path between the cache where the data was found and the cache that issued the request. A new TTL is set for each new copy of the data and a new TTL is also set at the cache, if any, where the data was found (by convention, the TTL at the server is infinite). This is in contrast with the model in [14] where there is no TTL reset upon a cache hit (new TTL is set only upon a cache miss). Resetting the TTL also at each cache hit increases the occupancy and the hit probability specially for popular contents (high λ). This choice is motivated by the CCN paradigm of moving popular documents as close as possible to the users.

We point out that in this description a request is instantaneously satisfied whether the data is found locally, at a remote cache or at a server. This corresponds to a situation where transmission times are negligible with respect to the frequency at which the data is requested. We do so since our primary objective is to investigate the traffic generated in the network in response to a request for the data.

We define the *miss process* at a cache as the successive time instants at which misses occur at this cache, namely, the times at which the data is requested and is not found in the cache. Let us denote by $\mathcal{C}(n)$ the set of children of cache n. The (overall) *request process*, also called the *arrival process*, at cache n is the superposition of the miss processes of caches in $\mathcal{C}(n)$ and of the exogenous request process at cache n, if any. We assume that successive TTLs at each cache are iid rvs, that TTLs at different caches are mutually independent, and that all TTLs are independent of the exogenous arrivals.

λ	Arrival rate (single cache)
$1/\mu$	Expected TTL (single cache)
$F(t)$	CDF exogenous arrivals (single cache)
$G(t)$	CDF inter-miss times (single cache)
$T(t)$	CDF TTL duration (single cache)
λ_n	Exogenous arrival rate at cache n
Λ_n	Overall arrival rate at cache n
$\Lambda_{n,k}$	Overall arrival rate at cache n for content k
$1/\mu_n$	Expected TTL at cache n
$F_n(t)$	CDF exogenous arrivals at cache n
$H_n(t)$	CDF overall arrivals at cache n
$H_{n,k}(t)$	CDF of overall arrivals at cache n for content k
$G_n(t)$	CDF inter-miss times at cache n
$T_n(t)$	CDF TTL duration at cache n
$h_{P,n}, m_{P,n}$	Hit, miss probability resp. at cache n
$h_{R,n}, m_{R,n}$	Hit, miss rate resp. at cache n
π_n	Occupancy of cache n (stationary probability content is in cache n)
$\pi_{n,k}$	Occupancy of cache n for content k
q_n	Average size of cache n ($= \pi_n$ if single content in network)
h_P, m_P	Hit, miss probability resp. (single cache)
h_R, m_R	Hit, miss rate resp. (single cache)
$\mathcal{C}(n)$	Set of children of cache n
$\chi^*(s)$	LST of CDF $\chi(t)$

Table 1: Glossary of main notation

Throughout the paper $h_{P,n}$ (resp. $m_{P,n} = 1 - h_{P,n}$) and $h_{R,n}$ (resp. $m_{R,n}$) denote the stationary hit (resp. miss) probability and the stationary hit (resp. miss) rate at cache n, respectively. We denote by π_n the steady-state probability that the data is in cache n and we call it the *occupancy* of cache n.

If Λ_n is the arrival rate of requests at cache n, the hit rate is given by $h_{R,n} = \Lambda_n h_{P,n}$ and the miss rate by $m_{R,n} = \Lambda_n(1 - h_{P,n})$. As a result once we have calculated $h_{P,n}$ and Λ_n also the hit/miss probability and hit/miss rate at cache n are determined.

For any non-negative rv X with Cumulative Distribution Function (CDF) $\chi(t) = P(X < t)$ $(t \geq 0)$, $\chi^\star(s) = E[e^{-sX}] = \int_0^\infty e^{-st} d\chi(t)$ $(s \geq 0)$ denotes its Laplace-Stieltjes Transform (LST), and $\hat{\chi}(s) = \int_0^\infty e^{-st}\chi(t)dt$ $(s > 0)$ denotes the Laplace Transform (LT) of $\chi(t)$. The identity

$$\chi^*(s) = s\hat{\chi}(s), \quad s > 0, \tag{1}$$

will be extensively used throughout.

For any number $a \in [0, 1]$, $\bar{a} := 1 - a$. In particular, if $\chi(t)$ is a CDF, $\bar{\chi}(t) = 1 - \chi(t)$ is the corresponding Complementary Cumulative Distribution Function (CCDF).

Additional notation and definitions will be given when studying specific cache networks.

The following result, taken from [19, Formula (4.1)], will be repeatedly used throughout this paper .

Theorem 2.1 *The CCDF, denoted by $\bar{R}(t)$, of the inter-event times of the point process resulting from the superposition of K mutually independent renewal processes, labeled, $1, \ldots, K$, is given*

by

$$\bar{R}(t) = \sum_{k=1}^{K} \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} \bar{R}_k(t) \prod_{j=1, j\neq k}^{K} \alpha_j \int_t^{\infty} \bar{R}_j(u)du, \tag{2}$$

with $R_k(t)$ and $\alpha_k > 0$ the CDF of the inter-event times and the arrival rate of process k, respectively.

We observe that such a superposition is not in general a renewal process itself.

3 Exact results

3.1 Single cache with renewal arrivals and general TTLs

We consider a single TTL cache. Requests arrive at the cache according to a renewal process. Without loss of generality, we assume that the first request arrives at time $t = 0$ and finds an empty cache. We denote by X a generic inter-arrival time with CDF $F(t) = P(X < t)$ and density $f(t) = dF/dt$. We also denote by T a generic TTL duration, with CDF $T(t) = P(T < t)$. Since successive inter-arrival times and successive TTLs form two independent renewal sequences and since a miss triggers a new TTL, miss times are regeneration points of the state of the cache. This implies that miss times form a renewal process, with generic inter-miss time denoted by Y and CDF $G(t) = P(Y < t)$.

The stationary hit probability, hit rate and miss rate denoted by h_P, h_R and m_R, respectively, are given by

$$h_P = P(X \leq T) = \int_0^{\infty} F(t)dT(t), \tag{3}$$

$$h_R = \lambda h_P, \quad m_R = \lambda(1 - h_P) \tag{4}$$

respectively, with $\lambda := 1/E[X]$ the arrival rate. The miss rate is alternatively given by $m_R = 1/E[Y]$ and the hit rate by $h_R = \lambda - m_R$.

Proposition 3.1 gives the cache occupancy π (i.e. the steady-state probability that the content is in the cache), for arbitrary inter-arrival time and TTL distributions.

Proposition 3.1 (Stationary cache occupancy)

$$\pi := \lambda E\left[\int_0^X (1 - T(t))dt\right]. \tag{5}$$

Proof. Let V be the time during which the document is in the cache between two consecutive request arrivals. We have $\pi = E[V]/E[X] = \lambda E[V]$ by renewal theory. Let us find $E[V]$. Define the binary rv $U(t)$ to be one if the document is in the cache at time t and zero otherwise. Without loss of generality consider the interval $[0, X]$ corresponding to the inter-arrival time between the first and the second request. We have

$$E[V] = E\left[\int_0^X U(t)dt\right] = E_X\left[\int_0^X E[U(t)|X]dt\right] = E_X\left[\int_0^X (1 - T(t))dt\right]$$

where the last equality follows from $E[U(t)|X] = E[U(t)] = P(U(t) = 1) = P(T > t)$. ◇

If TTLs are exponentially distributed with rate μ

$$\pi = \frac{\lambda(1 - F^*(\mu))}{\mu} \tag{6}$$

from (5). If arrivals are Poisson (with rate λ) $\pi = 1 - T^*(\lambda)$. If arrivals are Poisson and TTLs are exponentially distributed with rate μ then $\pi = \lambda/(\lambda + \mu)$. Note that $h_P = \pi$ if arrivals are Poisson, thanks to the PASTA property.

The proposition below provides a way for calculating $G(t)$, the CDF of the inter-miss time, a quantity that we will need later on.

Proposition 3.2 . *The CDF of inter-miss times is the unique bounded solution of the integral equation*

$$G(t) = \int_0^t G(t-x)(1 - T(x))dF(x) + \int_0^t T(x)dF(x). \tag{7}$$

Proof. Let X_1 (resp. Y_1, T_1) denote the first inter-arrival time (resp. first inter-miss time, first TTL) after $t = 0$. Since $Y_1 \geq X_1$, the event $\{Y_1 < t\}$ may only occur if $X_1 < t$. Therefore,

$$\begin{aligned} G(t) &= P(Y_1 < t, X_1 < t, X_1 \leq T_1) + P(Y_1 < t, T_1 < X_1 < t) \\ &= P(Y_1 < t, X_1 < t, X_1 \leq T_1) + P(T_1 < X_1 < t) \qquad (8) \\ &= P(Y_1 < t, X_1 < t, X_1 \leq T_1) + \int_0^t T(x)dF(x) \qquad (9) \end{aligned}$$

where (8) follows from the fact that the event $\{Y_1 < t\}$ is true when $T_1 < X_1 < t$. It remains to evaluate the probability $P(Y_1 < t, X_1 < t, X_1 \leq T_1)$ in (9). By conditioning on X_1 and T_1 we obtain

$$\begin{aligned} P(Y_1 < t, X_1 < t, X_1 \leq T_1) &= \int_{x=0}^t \int_{\tau=x}^\infty G(t-x)dF(x)dT(\tau) \\ &= \int_0^t G(t-x)(1 - T(x))dF(x), \qquad (10) \end{aligned}$$

where the first equality is due to the fact that the TTL is renewed at each request and then $Y_1 - X_1$ conditioned to $X_1 \leq T_1$ has the same distribution of Y_1.

Suppose that there are two solutions $G_1(t)$ and $G_2(t)$ satisfying (7). Then $G_1(t) - G_2(t) = \int_0^t (G_1(t) - G_2(t))(1 - T(x))dF(x)$. By Laplace transforming both sides of this equality, it appears evident that $G_1^*(s) - G_2^*(s) = 0$ and then the solution is unique. ◇

Define $h(t) = f(t)T(t)$, where we recall that $f(t)$ is the density of $F(t)$. Taking the Laplace transform of both sides of (7) yields

$$\hat{G}(s) = (F^*(s) - \hat{h}(s))\hat{G}(s) + \frac{\hat{h}(s)}{s}$$

from which we get

$$\hat{G}(s) = \frac{\hat{h}(s)}{s(1 - F^*(s) + \hat{h}(s))} \tag{11}$$

and, from (1),

$$G^*(s) = \frac{\hat{h}(s)}{1 - F^*(s) + \hat{h}(s)}. \tag{12}$$

If the TTL is a constant, equal to T, (7) becomes

$$G(t) = \int_0^{t\wedge T} G(t-x)dF(x) + (F(t) - F(T))\mathbf{1}(t > T) \tag{13}$$

with $a \wedge b = \min(a, b)$. In this case the hit probability is $F(T)$.

We conclude this section by examining two particular cases:

Example 1: Exponentially distributed TTLs For $T(t) = 1 - e^{-\mu t}$, $h_P = F^*(\mu)$ from (3), which in turn implies from (4) that $h_R = \lambda F^*(\mu)$ and $m_R = \lambda(1 - F^*(\mu))$. On the other hand, $\hat{h}(s) = F^*(s) - F^*(s + \mu)$ so that (12) becomes

$$G^*(s) = \frac{F^*(s) - F^*(s+\mu)}{1 - F^*(s+\mu)}. \tag{14}$$

The miss rate can also be obtained from (14) as $m_R = 1/E[Y] = -1/dG^*(s)/ds|_{s=0} = \lambda(1 - F^*(\mu))$, considering that $-dF^*/ds|_{s=0} = 1/\lambda$ and $F^*(0) = 1$.

Example 2: Poisson arrivals, exponentially distributed TTLs Let $T(t) = 1 - e^{-\mu t}$ (exponential TTLs) and $F(t) = 1 - e^{-\lambda t}$ (Poisson requests with arrival rate λ). From Eqs. (3)-(4) we find $h_P = F^*(\mu) = \lambda/(\lambda + \mu)$, $h_R = \lambda^2/(\lambda + \mu)$ and $m_R = \lambda\mu/(\lambda + \mu)$. The LST of Y is given by (use (14) with $F^*(s) = \lambda/(s + \lambda)$):

$$G^*(s) = \frac{\lambda}{\lambda + s} \times \frac{\mu}{\mu + s} \tag{15}$$

which shows that the inter-miss time is the sum of two independent exponential rvs with parameters λ and μ, with CDF

$$G(t) = \begin{cases} 1 - \frac{\mu e^{-\lambda t} - \lambda e^{-\mu t}}{\mu - \lambda} & \text{if } \lambda \neq \mu \\ 1 - (1 + \lambda t)e^{-\lambda t} & \text{if } \lambda = \mu. \end{cases} \tag{16}$$

This result is not surprising since, in the case of exponential TTLs, it is equvalent to regenerate or not the TTL at each hit, and then the inter-miss time is equal to the sum of a TTL (an exponential rv with rate μ) and of the time until the first request arrival after the expiration of the TTL (an exponential rv with rate λ). In particular,

$$m_R = \frac{1}{E[Y]} = \frac{1}{-\frac{dG^*(s)}{ds}|_{s=0}} = \frac{\lambda\mu}{\lambda + \mu} = \frac{1}{E[T] + E[X]}. \tag{17}$$

3.2 Line of caches with exogenous arrivals at a single cache and exponential TTLs

Consider the line network in Fig. 1 composed of N TTL caches labeled $1, \ldots, N$, with no exogenous requests submitted at caches $2, \ldots, N$. Requests arrive to cache 1 according to a renewal process with generic inter-arrival time X and arrival rate λ. According to the description made at the beginning of Section 2, upon a miss at cache 1 the first cache to hold the document, say cache $n \leq N$, returns a copy of the document to caches $n - 1, \ldots, 1$ and all TTLs are reinitialized. TTLs at all caches are mutually independent and *exponentially* distributed rvs with rate μ_n at cache n.

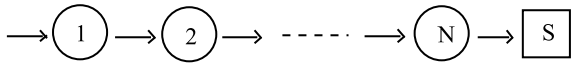


Figure 1: Line of caches with exogenous arrivals only at cache 1 (S = server)

The arrival process at cache n is the miss process at cache $n-1$ since there are no exogenous arrivals. Morever, it is easily seen that the miss times at cache $n-1$ form regeneration points, so that the miss process at this cache, and therefore the arrival process at cache n, is a renewal process. We can then apply recursively the results obtained for a single cache. In particular, if we denote by $G_n^*(s)$ the LST of the inter-miss times at cache n, we may apply formula (14) where the LST of the interarrival times is G_{n-1}^*. We obtain:

$$G_n^*(s) = \frac{G_{n-1}^*(s) - G_{n-1}^*(s+\mu_n)}{1 - G_{n-1}^*(s+\mu_n)} \tag{18}$$

for $n = 1, \ldots, N$, where $G_0^*(s) = F^*(s)$. The hit probability at cache n is given by $h_{P,n} = G_{n-1}^*(\mu_n)$, since a hit occurs at cache n if the time duration between two successive requests at this cache (with CDF $G_{n-1}(t)$) does not exceed the TTL duration.

If we denote by $Y(n)$ the generic inter-miss time at cache n, the miss rate at this cache is given by

$$m_{R,n} = \frac{1}{E[Y(n)]} = \frac{1}{-dG_n^*(s)/ds|_{s=0}}, \tag{19}$$

so that by using (18)

$$m_{R,n} = m_{R,n-1}(1 - G_{n-1}^*(\mu_n)) = \lambda \prod_{i=0}^{n-1}(1 - G_i^*(\mu_{i+1})) \tag{20}$$

with $m_{R,0} := \lambda$ by convention. The hit rate at cache n is

$$h_{R,n} = m_{R,n-1}h_{P,n} = \lambda G_{n-1}^*(\mu_n) \prod_{i=0}^{n-2}(1 - G_i^*(\mu_{i+1})) \tag{21}$$

with $G_0(t) = F(t)$ by convention. The occupancy of cache n (i.e. the stationary probability that the content is at cache n) is given by (Hint: apply (6) with $F(t) = G_{n-1}(t)$ and $E[X] = 1/m_{R,n-1}$)

$$\pi_n = m_{R,n-1}\left(\frac{1 - G_{n-1}^*(\mu_n)}{\mu_n}\right). \tag{22}$$

The unknown constants $\{G_{n-1}^*(\mu_n)\}_{n=2}^N$ in (20)-(22) can be recursively computed from (18).

3.3 Simple tree network with Poisson exogenous arrivals and exponential TTLs

Consider the tree network in Fig. 2 with one root (labeled $N+1$) and N children (leaves) labeled $1, \ldots, N$. Exogenous requests arrive at cache $n = 1, \ldots, N+1$ according to a Poisson process with rate λ_n. Caches $1, \ldots, N$ have exponential distributed TTLs with rate μ_n at cache n. We assume that TTLs at cache $N+1$ have an arbitrary CDF $T_{N+1}(t)$, with LST $T_{N+1}^*(s)$. For each $n = 1, \ldots, N$, cache n behaves as a TTL cache in isolation, so that the cache occupancy, the

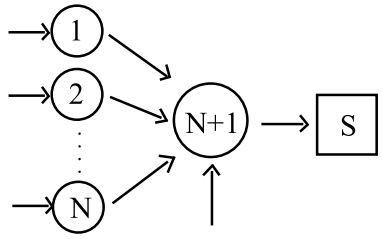


Figure 2: Simple tree network

hit/miss probability, and the hit/miss rate at this cache are given in Section 3.1 (with $\lambda = \lambda_n$ and $\mu = \mu_n$). The total arrival rate at cache n is $\Lambda_n = \lambda_n$ for $n = 1, \ldots, N$, and the total arrival rate at cache $N+1$ is

$$\Lambda_{N+1} = \lambda_{N+1} + \sum_{i=1}^{N} \nu_i \tag{23}$$

with $\nu_i = \lambda_i \mu_i / (\lambda_i + \mu_i)$ from (17).

Let us focus on cache $N+1$. The arrival process at cache $N+1$ is the superposition of a Poisson process with rate λ_{N+1} and the mutually independent miss renewal processes of caches $1, \ldots, N$, with the CDF of the inter-miss times at cache $n = 1, \ldots, N$ given by (use (16))

$$G_n(t) = \begin{cases} 1 - \frac{\mu_n e^{-\lambda_n t} - \lambda_n e^{-\mu_n t}}{\mu_n - \lambda_n} & \text{if } \lambda_n \neq \mu_n \\ 1 - (1 + \lambda_n t) e^{-\lambda_n t} & \text{if } \lambda_n = \mu_n. \end{cases}$$

The CDF, denoted by $H_{N+1}(t)$, of the overall inter-arrival times at cache $N+1$ is obtained by using Theorem 2.1. Tedious (but easy) algebra yields

$$H_{N+1}(t) = 1 - \frac{e^{-\lambda_{N+1} t}}{\Lambda_{N+1}} \left(\prod_{i=1}^{N} \frac{\mu_i^2 e^{-\lambda_i t} - \lambda_i^2 e^{-\mu_i t}}{\mu_i^2 - \lambda_i^2} \right) \left(\lambda_{N+1} + \sum_{i=1}^{N} \lambda_i \mu_i \frac{\mu_i e^{-\lambda_i t} - \lambda_i e^{-\mu_i t}}{\mu_i^2 e^{-\lambda_i t} - \lambda_i^2 e^{-\mu_i t}} \right). \tag{24}$$

With the help of the identity

$$\prod_{i=1}^{N} (\mu_i^2 e^{-\lambda_i t} - \lambda_i^2 e^{-\mu_i t}) = \prod_{i=1}^{N} (-\lambda_i^2) \sum_{\substack{i_l \in \{0,1\} \\ l=1,\ldots,N}} \prod_{k=1}^{N} \left((-1)^{i_k} \left(\frac{\mu_k}{\lambda_k} \right)^{2 i_k} \right) e^{-\left(\sum_{k=1}^{N} (\lambda_k^{i_k} + \mu_k^{1-i_k}) \right) t} \tag{25}$$

we can express $H_{N+1}(t)$ as a weighted sum of negative exponential terms, as shown in the appendix (see (43)). The hit probability at cache $N+1$ is given by (use (43))

$$\begin{aligned} h_{P,N+1} &= \int_0^\infty H_{N+1}(t) dT(t) \qquad (26) \\ &= 1 - \frac{(-1)^N}{\Lambda_{N+1}} \prod_{i=1}^{N} \frac{\lambda_i^2}{\mu_i^2 - \lambda_i^2} \Bigg[\lambda_{N+1} \sum_{\substack{i_l \in \{0,1\} \\ l=1,\ldots,N}} \prod_{k=1}^{N} (-1)^{i_k} \left(\frac{\mu_k}{\lambda_k} \right)^{2 i_k} \\ &\quad \times T^*_{N+1} \left(\lambda_{N+1} + \sum_{k=1}^{N} (\lambda_k^{i_k} + \mu_k^{1-i_k}) \right) \end{aligned}$$

$$-\sum_{i=1}^{N}\frac{\mu_i}{\lambda_i}\sum_{\substack{i_l\in\{0,1\}\\ l=1,\ldots,N,l\neq i}}\prod_{\substack{k=1\\k\neq i}}^{N}(-1)^{i_k}\left(\frac{\mu_k}{\lambda_k}\right)^{2i_k}$$
$$\times\left(\mu_i T^*_{N+1}(\lambda_{N+1}+\lambda_i+\sum_{\substack{k=1\\k\neq i}}^{N}(\lambda_k^{i_k}+\mu_k^{1-i_k}))\right.$$
$$\left.-\lambda_i T^*_{N+1}(\lambda_{N+1}+\mu_i+\sum_{\substack{k=1\\k\neq i}}^{N}(\lambda_k^{i_k}+\mu_k^{1-i_k}))\right)\Bigg]. \quad (27)$$

The hit rate (resp. miss rate) at cache $N+1$ is given by $h_{R,N+1}=\Lambda_{N+1}h_{P,N+1}$ (resp. $m_{R,N+1}=\Lambda_{N+1}(1-h_{P,N+1})$).

Example 3: Caches $1,\ldots,N$ identical Assume that $\lambda:=\lambda_n$ and $\mu:=\mu_n$ for $n=1,\ldots,N$. Eq. (24) reduces to

$$H_{N+1}(t)=1-\sum_{n=0}^{N}a_n e^{-c_n t}-\sum_{n=0}^{N-1}b_n\left(\mu e^{-c_{n+1}t}-\lambda e^{-c_n t}\right) \quad (28)$$

with

$$a_n:=\binom{N}{n}\frac{\lambda_{N+1}\mu^{2n}(-\lambda^2)^{N-n}}{\Lambda_{N+1}}\times\frac{1}{(\mu^2-\lambda^2)^N}$$
$$b_n:=\binom{N-1}{n}\frac{N\lambda\mu^{2n+1}(-\lambda^2)^{N-1-n}}{\Lambda_{N+1}}\times\frac{1}{(\mu^2-\lambda^2)^N}$$
$$c_n:=\lambda n+\mu(N-n)+\lambda_{N+1}.$$

The hit probability at cache $N+1$ is (use (26))

$$h_{P,N+1}=1-\sum_{n=0}^{N}a_n T^*_{N+1}(c_n)-\sum_{n=0}^{N-1}b_n\left(\mu T^*_{N+1}(c_{n+1})-\lambda T^*_{N+1}(c_n)\right). \quad (29)$$

4 Approximate results

The exact results in Section 3 cannot be easily extended to general networks. The main problem is that when the aggregate arrival process at a cache is not a renewal process, we can still determine the main performance metrics at the cache if we can calculate the CDF of the inter-arrival times, but we cannot apply Proposition 3.2 that allows us to characterize the miss rate and then to study the upstream caches.

In this section we develop an approximation method which produces highly accurate approximations under more general topologies for all metrics considered in Section 3 (hit/miss probabilities, hit/miss rate, cache occupancy). The quality of the approximation is assessed in Section 5.

Our approximation is based on the following assumption:

Assumption A1: the overall arrival process at each node is a renewal process.

A direct consequence of Assumption **A1** is that we approximate also the miss process at a node as a renewal process.

With a slight abuse of notation we will use the notation of Section 3 to denote the corresponding approximate values calculated under Assumption **A1**. For example $H_n(t)$ is used to

denote the approximate CDF of the overall inter-arrival times. Similarly $G_n(t)$, Λ_n, $m_{R,n}$, $h_{P,n}$ and $h_{R,n}$ are used to denote approximate quantities. Regarding the total rate Λ_n, note that

$$\Lambda_n = \lambda_n + \sum_{i \in \mathcal{C}(n)} m_{R,i} \tag{30}$$

where $\mathcal{C}(n)$ is the set of children of node n. As in Section 3, exogenous arrivals at each node form a renewal process, with CDF $F_n(t)$ at node n.

Assumption **A1** allows us to invoke Theorem 2.1 to get

$$H_n(t) = 1 - \frac{\lambda_n}{\Lambda_n} \bar{F}_n(t) \prod_{i \in \mathcal{C}(n)} \nu_i \int_t^\infty \bar{G}_i(u)du - \sum_{i \in \mathcal{C}(n)} \frac{\nu_i}{\Lambda_n} \bar{G}_i(t) \lambda_n \int_t^\infty \bar{F}_n(u)du \prod_{\substack{j \in \mathcal{C}(n) \\ j \neq i}} \nu_j \int_t^\infty \bar{G}_j(u)du. \tag{31}$$

An approximation of the CDF of the inter-miss times at cache n is obtained from Proposition 3.2

$$G_n(t) = \int_0^t G_n(t-x)(1 - T_n(x))dH_n(x) + \int_0^t T_n(x)dF_n(x) \tag{32}$$

where $T_n(t)$ the CDF of the TTL duration at cache n.

Eqs (31)-(32) provide a recursive procedure for calculating, at least numerically, approximations for the CDFs $G_n(t)$ and $H_n(t)$ for each cache n, from which we can derive approximate formulas for the hit/miss probability, the hit/miss rate, and the occupancy at each cache. In particular, for a general tree network the procedure requires calculating the CDFs for all the caches at the same depth, starting from those farthest from the root. The inter-miss time CDF at a given leaf cache can be derived from the exogenous request process at the cache through renewal equation (32). For a cache at the level above the inter-arrival request CDF can be calculated using Eq. (31) to combine the CDFs of the inter-arrival times of its exogenous request process and the inter-miss times for its children. We can apply again the renewal equation to characterize the output process at this cache and so on.

However, even for small networks the numerical complexity of this procedure can be very high as it requires solving integral equations (see Eq. (32)) and calculating integrals over infinite ranges (see Eq. (31)).

In order to get explicit results we now focus on a particular class of tree networks, class $\mathcal{N}$. TTLs at each node are exponentially distributed with rate μ_n. A network belongs to class $\mathcal{N}$ if, in addition to assumption **A1**, the following assumption holds:

Assumption A2: For each n, node n is fed by the superposition of two independent request arrival processes: one (*stream 1*) is the miss rate of a child of cache n and is a generic renewal process and the other one (*stream 2*) is a renewal process with CDF of the form

$$K_n(t) = 1 - \sum_{m=0}^{M_n} \alpha_{n,m} e^{-\beta_{n,m} t} \tag{33}$$

where $0 \leq M_n < \infty$ and $\{\beta_{n,m}\}_m$ is a set of nonnegative numbers.

In what follows we assume without loss of generality that stream 1 originates from a cache child labeled $n-1$ and then we denote the CDF of the inter-miss times in stream 1 as $G_{n-1}(t)$ and the miss rate as ν_{n-1}. Since

$$\eta_n := \sum_{m=0}^{M_n} \alpha_{n,m} \beta_{n,m} \tag{34}$$

is the arrival rate of stream 2 from (33), the total arrival rate at node n is

$$\Lambda_n = \nu_{n-1} + \eta_n. \tag{35}$$

Assumptions **A1** and **A2** together yield the following procedure for approximating $G_n^*(s)$ and $H_n^*(s)$.

Proposition 4.1 (Approximation for class $\mathcal{N}$)
Under Assumptions **A1** *and* **A2**, *for each node* n,

$$H_n^*(s) = 1 - s\frac{\eta_n}{\Lambda_n}\sum_{m=0}^{M_n}\frac{\alpha_{n,m}}{s+\beta_{n,m}} - s^2\frac{\eta_n\nu_{n-1}}{\Lambda_n}\sum_{m=0}^{M_n}\frac{\alpha_{n,m}(1-G_{n-1}^*(s+\beta_{n,m}))}{(s+\beta_{n,m})^2\beta_{n,m}} \tag{36}$$

and

$$G_n^*(s) = \frac{H_n^*(s) - H_n^*(s+\mu_n)}{1 - H_n^*(s+\mu_n)}. \tag{37}$$

Proof.

$$H_n(t) = 1 - \frac{\eta_n\nu_{n-1}}{\Lambda_n}\sum_{m=0}^{M_n}\alpha_{n,m}\left(\bar{G}_{n-1}(t)\int_t^\infty e^{-\beta_{n,m}u}du + e^{-\beta_{n,m}t}\int_t^\infty \bar{G}_{n-1}(u)du\right)$$

from which we deduce (36). (37) is obtained from (14). ◇

Differentiating both sides of (37) wrt to s and letting $s = 0$ yields

$$\nu_n = \Lambda_n(1 - H_n^*(\mu_n)) = (\nu_{n-1} + \eta_n)(1 - H_n^*(\mu_n)) \tag{38}$$

where the second equality comes from (35) (Hint: $\Lambda_n = -(dH_n^*(s)/ds|_{s=0})^{-1}$), so that

$$\nu_n = \sum_{i=1}^{n}\eta_i\prod_{j=i}^{n}(1 - H_j^*(\mu_j)) \tag{39}$$

and, finally (using (35))

$$\Lambda_n = \sum_{i=1}^{n-1}\eta_i\prod_{j=i}^{n-1}(1 - H_j^*(\mu_j)) + \eta_n. \tag{40}$$

Observe that the first equality in (38) can be obtained without any calculation since $H_n^*(\mu_n)$ is the hit probability at node n so that $\Lambda_n(1 - H_n^*(\mu_n))$ is the miss rate at node n.

Relations (36)-(37) and (39)-(40) provide a recursive procedure for calculating Λ_n and $H_n^*(\mu_n)$ for each n, from which we obtain approximations for the hit probability, hit rate, miss rate and stationary occupancy at node n:

$$\begin{aligned} h_{P,n} &= H_n^*(\mu_n),\ h_{R,n} = \Lambda_n H_n^*(\mu_n), \\ m_{R,n} &= \Lambda_n(1 - H_n^*(\mu_n)),\ \pi_n = \Lambda_n\left(\frac{1 - H_n^*(\mu_n)}{\mu_n}\right). \end{aligned} \tag{41}$$

The latter result follows from (6).

Below, we present two networks belonging to the class $\mathcal{N}$, i.e. with exponentially distributed TTLs and satisfying assumption **A2**.

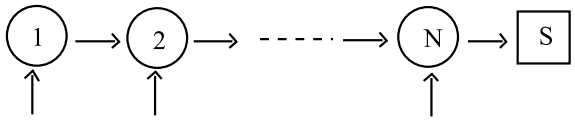


Figure 3: Line of caches with exogenous arrivals

4.1 Line of N TTL caches with Poisson exogenous arrivals and exponential TTLs

This is the same line network as in Section 3.2 with the addition of Poisson exogenous arrivals at all caches $1, \ldots, N$ with rate λ_n at cache n, see Fig. 3.

This network belongs to the class $\mathcal{N}$ with $M_n = 0$, $\alpha_{n,0} = 1$, $\beta_{n,0} = \lambda_n$ for each n, so that $\eta_n = \lambda_n$ and

$$\Lambda_n = \sum_{i=1}^{n-1} \lambda_i \prod_{j=i}^{n-1} (1 - H_j^*(\mu_j)) + \lambda_n$$

from (34) and (40).

Remark 4.1 (Exact results at nodes 1 and 2)
Quantities in the r.h.s. of (41) give the exact hit probability, hit rate, miss rate, and occupancy at node $n = 1, 2$ for the line network in Fig. 3 since caches 1 and 2 form a simple tree network for which our analysis is exact.

4.2 Line of simple tree networks

Consider the network in Fig. 4: node n is fed by node $n-1$ and by the superposition of the miss processes of single caches (node 1 is only fed by single caches; the analysis below extends to the case where node 1 is fed by an additional Poisson stream of requests). For the sake of simplicity, here we assume that there are R_n *identical* TTL caches feeding node n, but the analysis can be extended to the heterogeneous case (see below). Each of these single caches is fed by an exogenous stream of Poisson requests with rate δ_n and has exponentially distributed TTLs with rate γ_n. Nodes $1, \ldots, N$ have exponential TTLs with rate μ_n at node n. Denote by $\mathcal{S}(n)$ the set of R_n single TTL caches associated with cache n. Let us show that this network belongs to $\mathcal{N}$.

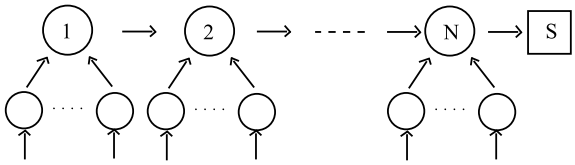


Figure 4: Line of simple tree networks

We need to show that the CDF of the inter-arrival times of requests joining node n from caches in $\mathcal{S}(n)$ (denoted by $K_n(t)$) has the form (33). This CDF has already been calculated in Section 3.3 – use formula (28) with $N = R_n$, $\lambda_{N+1} = 0$, $\lambda = \delta_n$, $\mu = \gamma_n$, $\Lambda_{N+1} = R_n \delta_n \gamma_n / (\delta_n + \gamma_n)$. It is then easily seen that (33) holds with

$$M_n = R_n$$

$$\begin{aligned}
\beta_{n,m} &= \delta_n m + \gamma_n(R_n - m) \\
\alpha_{n,0} &= -\delta_n b_{n,0} \\
\alpha_{n,m} &= \gamma_n b_{n,m-1} - \delta_n b_{n,m}, \quad m = 1, \ldots, R_n - 1 \\
\alpha_{n,R_n} &= b_{n,R_n-1}\gamma_n
\end{aligned}$$

where

$$b_{n,m} := \binom{R_n - 1}{m} \frac{\gamma_n^{2m}(-\delta_m^2)^{R_n-1-m}(\delta_m + \gamma_m)}{(\gamma_n^2 - \delta_n^2)^{R_n}}$$

for $m = 0, \ldots, R_n$. A similar analysis can be carried out when, for every n, the R_n caches feeding node n are not identical. In this case, formula (43) in Appendix should be used instead of (28).

Remark 4.2 (Exact results at node 1 & leaves) *Quantities in the r.h.s. of (41) give the exact hit probability, hit rate, miss rate and occupancy at node 1 and at all leaf nodes of the network in Fig. 4. In fact our analysis is exact for a single cache and for a simple tree network (as that formed by node 1 and its children).*

5 Validation

In this section we investigate the accuracy of the approximation method developed in Section 4. Recall that the method consists in assuming that all internal arrival processes at a node (i.e. processes formed of the miss processes of the node's children) are renewal processes and to use Eq. (31) to calculate the CDF of the inter-arrival times of the superposed process. The miss process at this node can then be characterized by using Eq. (32) and the procedure is repeated at the node's parent.

We focus our validation on the case when the TTLs are exponentially distributed, but we also provide some results for constant TTLs.

5.1 Exponential timers

We start by observing that it is possible to model a class $\mathcal{N}$ network with N caches as an irreducible Markov process, with state $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t)) \in \mathcal{E} = \{0,1\}^N$, where $x_n(t) = 1$ (resp. $x_n(t) = 0$) if the document is present (resp. missing) at time t at node n. Once the steady-state probabilities ($p(\mathbf{x})$) have been calculated, the exact values of the performance metrics of interest can be obtained by conveniently combining the stationary probabilities and the rates. For example the stationary occupancy of cache i is $\pi_i^M = \sum_{\mathbf{x}\in\mathcal{E}, x_i=1} p(\mathbf{x})$ (the superscript "M" stands for "Markov") . For a line of caches the hit probability and the miss rate at cache 1 are respectively $h_{P,1}^M(1) = p(1,*)$ and $m_{R,1}^M = \lambda_1 p(0,*)$, while for cache 2 it holds

$$h_{P,2}^M = \frac{\lambda_1 p(0,1,*) + \lambda_2(p(0,1,*) + p(1,1,*))}{\lambda_1(p(0,0,*) + p(0,1,*)) + \lambda_2}$$

and $m_{R,2}^M = \lambda_1 p(0,0,*) + \lambda_2(p(0,0,*) + p(1,0,*))$, where $p(i,*) = \sum_{x_2,\ldots,x_N\in\{0,1\}} p(i, x_2, \ldots, x_N)$ and $p(i,j,*) := \sum_{x_3,\ldots,x_N\in\{0,1\}} p(i,j,x_3,\ldots,x_N)$ are the stationary probabilities that cache 1 is in state $i \in \{0,1\}$ and caches $(1,2)$ are in state $(i,j) \in \{0,1\}^2$, respectively. Due to space constraints we omit the general expressions for these quantities for a generic cache in the line and those for a line of simple tree networks that can be similarly calculated.

In the rest of this section we compare our approximate results versus the exact ones that can be obtained studying the Markov process. A comparison of the computational costs of

the two approaches is in Section 6. We consider first the line network in Fig. 5: it has four nodes ($N = 4$), exogenous arrivals and exponentially distributed TTLs at node n with rate λ_n and μ_n, respectively. We have calculated the absolute relative errors at cache n for the hit

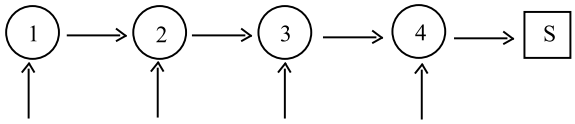


Figure 5: Line of four caches

probability ($E_{HP,n}$), the miss rate ($E_{MR,n}$) and the occupancy probability ($E_{OP,n}$). The exact value is calculated through the analysis of the Markov process, e.g. $E_{HP,n} := |h^M_{P,n} - h_{P,n}|/h^M_{P,n}$. Fig. 6 shows the CDFs of the relative errors at cache 4 for 1001 different parameter vectors $((\lambda_n, \mu_n), n = 1, \ldots, 4)$. The values of the exogeneous arrival rates (resp. TTL rates) have been selected in the interval $[0.001, 10]$ (resp. $[0.1, 2]$) according to the FAST (Fourier Amplitude Sensitivity Test) method (see [18, Sec. VI-C] and references therein). We can observe that the approximation is very accurate: in 99% of the different parameter settings the relative error is smaller than $2 * 10^{-5}$.

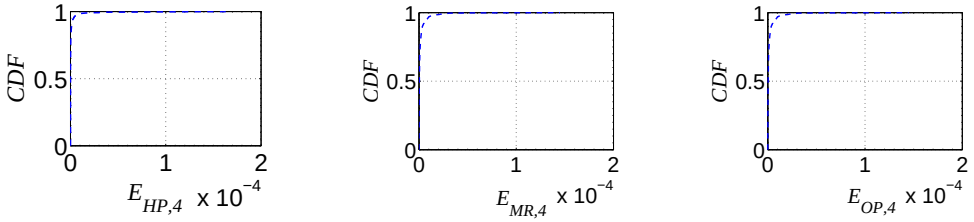


Figure 6: CDF of $E_{HP,4}, E_{MR,4}, E_{OP,4}$ for network in Fig.5

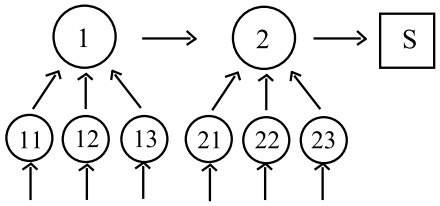


Figure 7: Linear tree network

Fig. 8 shows how the error changes for different request loads. In this case we have considered the homogeneous scenario where all the caches have the same TTL and the same exogenous arrival rate, i.e. $\mu_n = \mu$ and $\lambda_n = \lambda$ for each n. The error is shown as a function of the normalized load $\rho = \lambda/\mu$. We can observe that the largest error (about 2×10^{-4}) is obtained when arrival rates and timer rates have comparable values ($\rho \approx 1$). In this case the different request processes superposed at a node have similar time scales and then the inter-arrival times of the overall request process are more correlated (see also comments below).

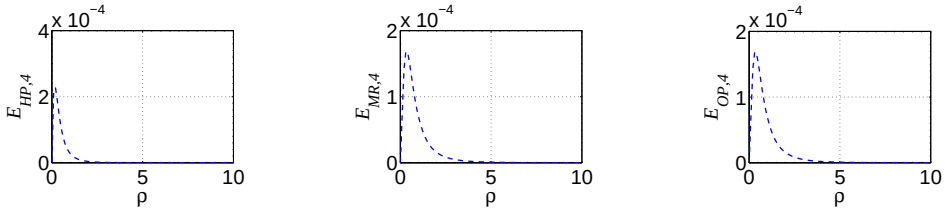


Figure 8: $E_{HP,4}, E_{MR,4}, E_{OP,4}$ for network in Fig. 7 with homogeneous nodes ($\lambda_n = \lambda = \rho\mu = \rho\mu_n$)

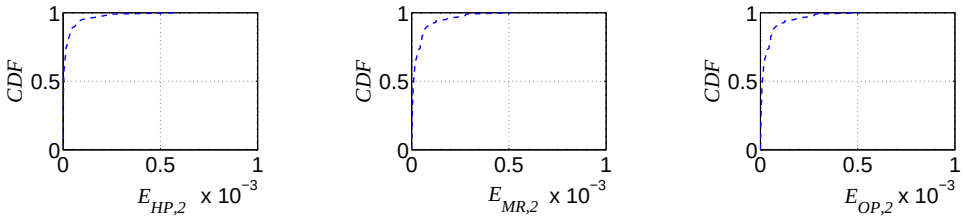


Figure 9: CDF of $E_{HP,2}$, $E_{MR,2}$, $E_{OP,2}$ for network in Fig. 7

We have also investigated the accuracy of the approximation for the line of simple tree networks (defined in Section 4.2) shown in Fig. 7: where nodes $11, 12, 13$ (resp. nodes $21, 22, 23$) have identical arrival rates and identical TTLs rates. Since the approximation results are exact for all nodes but node 2 we only report results for that node. The empirical CDFs of $E_{HP,2}$, $E_{MR,2}$ and $E_{OP,2}$ are shown in Fig. 9. Like for Fig. 6 exact results have been obtained by considering the Markov process associated with this line of simple trees network. Different request and TTL rates have been selected according to the FAST method respectively in the intervals $[0.001, 10]$ and $[0.1, 2]$. We used 4921 samples for each rate. Results are analogous to those for a line of caches. The relative errors can be larger in this scenario, but they are probably negligible for most of the applications ($< 3 \times 10^{-4}$ in 99% of the cases). We have also considered the homogenous scenario also for this topology, the relative errors have the same order of magnitude ($< 10^{-3}$).

We have shown that Assumption **A1** leads to very accurate results when exogenous arrival processes are Poisson and TTL are exponentially distributed. This let us think that the superposition of the request arrival processes at every cache is very 'close' to a renewal process. In order to justify such statement, we have calculated the first autocorrelation lag (r_1) for the actual arrival process at node 2 in Fig. 5 using Eq. (6.4) in [19]. This autocorrelation lag depends on the arrival rates λ_1 and λ_2 and the timer μ_1. We have found that for any possible choice of these parameters $0 > r_1 > -0.015$. Simulation results show that the autocorrelation is even less significant at larger lags. We can then conclude that the inter-arrival times are weakly coupled.

5.2 Deterministic Timers

When timers are deterministic we need to rely on the general procedure described in Section 4 and based on Eqs (31) and (32). There are two sources of errors in this procedure. Firstly, the aggregate request process at a cache is not a renewal process and it is not correct to apply the renewal equation (32). Secondly, both the steps (31) and (32) introduce some numerical

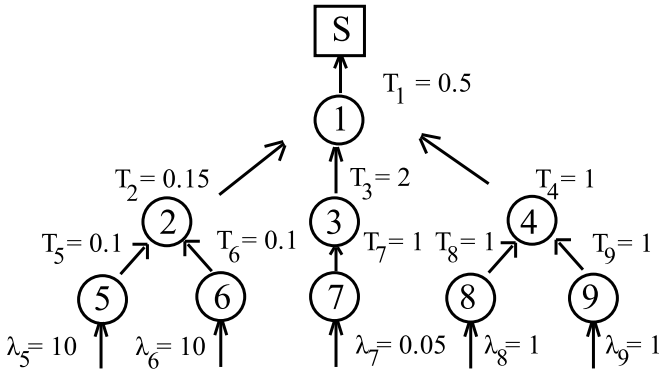


Figure 10: Tree network

errors. Two parameters determine the entity of the numerical error: 1) the time interval (τ) from which the CDF samples are taken, 2) the time distance between two consecutive samples (Δ). Clearly the larger τ and the smaller Δ the smaller the numerical error, but also the larger the computational cost.

We have implemented a Matlab numerical solver that iteratively determines the CDFs in the network as described above, and then the metrics of interest for a cache network. The integrals appearing in Eqs. (31) and (32) are approximated as simple sums and for simplicity the same values τ and Δ have been considered for all the CDFs. These parameters are selected as follows: our solver first obtains an approximated solution for the whole network assuming that all the request processes are Poisson and set the parameter τ to 5 times the largest expected inter-arrival time in the network. The parameter Δ is set to one thousandth of the minimum of the TTL values and the expected interarrival times of the exogenous request processes.

We present some preliminary results for the tree network in Fig. 10. The exogenous request processes are Poisson processes with rates λ_i ($i = 5, 6, 7, 8, 9$). TTL values T_i ($i = 1, 2, \dots 9$) are shown in the figure. In order to evaluate the relative error of the estimated metrics, we have considered as correct values those obtained through a long simulation. For example if our method predicts the value $h_{P,n}$ for the hit probability rate at node n and the 99% confidence interval, calculated by simulation, is $[h^S_{P,n}-\epsilon, h^S_{P,n}+\epsilon]$, the relative error is calculated as $|h_{P,n}-h^S_{P,n}|/h^S_{P,n}$. The relative incertitude of the simulation ($\epsilon/h^S_{P,n}$) is at most 0.310^{-4}. For all the performance metrics and all the caches the relative error of our approach is less than 10^{-2}.

6 Computational Cost

In this section we perform a preliminary analysis of the computational cost of our approach.

We first address the case of a class $\mathcal{N}$ network, and in particular we consider a line of simple tree networks with N trees and M nodes in total as in Fig. 4. Since the computational cost for all the metrics is roughly the same, we focus here on the hit probability. In order to calculate the hit probability at one of the roots of the simple trees, say it cache n, we need to evaluate the LST $H^*_n(\mu_n)$ (Eq. (41)). This requires a number of operations proportional to the number of children of cache n (R_n) and the evaluation of the LST of the miss rate coming from cache $n-1$ in μ_n, i.e $G^*_{n-1}(\mu_n)$ (Eq. (36)). In turn $G^*_{n-1}(\mu_n)$ can be calculated evaluating $H^*_{n-1}(s)$ in two points (μ_n and $\mu_n + \mu_{n-1}$) (Eq. (37)) and so on recursively. This implies that the cost to calculate the hit

probability at cache n is $O(\alpha R_n + 2^n)$ for some constant α. When evaluating the hit probability at other caches the same LSTs needs to be evaluated, but in general at different points, then we have that the total cost is $O(\sum_{n=1}^{N} \alpha R_n + 2^n) = O(\alpha M + 2^N)$. Then, depending on the topology of the network, the cost can be mainly linear in the number of nodes (for a network with small depth, e.g. when there are a few trees each with a lot nodes) or exponential in the number of nodes (for a network with large depth, e.g. for the linear network in Fig. 3).

It is interesting to compare this cost with alternative approaches. For the line of simple tree networks, all the metrics can be exactly calculated solving a Markov process as we mentioned in Section 5. The size of the state space is 2^M, then the cost of determining the steady-state distribution by solving the linear equation system is $O(2^{3M})$ and this is much larger than the cost of our method $O(\alpha M + 2^N)$. A different approach is to obtain an approximated steady-state distribution of the Markov process using an iterative method. This approach takes advantage of the fact that most of the transition rates have value zero. In fact a state change is triggered by an exogenous request arrival at a cache that does not have the data or by a timer expiration at a cache with the data, i.e. from a given state we can only reach other M states. Then the number of non-zero rates is equal to $M2^M$ and each iteration of the method requires $O(M2^M)$ operations. The total cost of the iterative method is then $O(KM2^M)$, where K is the number of iterations until termination and depends on the spectral gap of the matrix used at each iteration and on the required precision, but in general we can expect $O(KM2^M) << O(2^{3M})$. Assuming that this is the case, we can observe that our method, even in the worst case of the linear network, is still more convenient than solving the Markov process, because $O(2^M) < O(KM2^M)$.

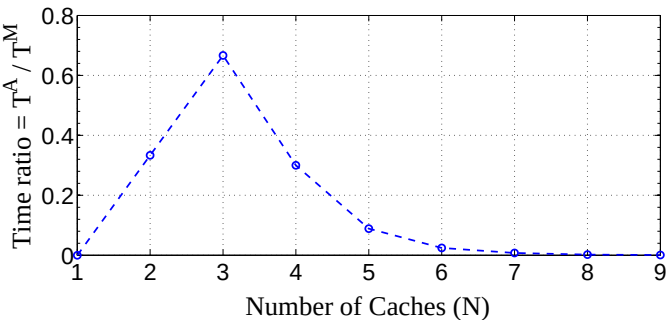


Figure 11: Running time comparison

Fig. 11 shows the ratio of the computation times to calculate our approximation (T^A) and to solve the Markov chain (T^M) for a line of N caches (with $N = 1, 2, \ldots 9$). Both the methods have been implemented in Matlab, in particular the function linsolve has been used to determine the steady-state distribution of the Markov chain.

Let us now consider the case of a general tree network with constant TTLs (equal to T). In this case there is no exact solution to compare our approach with, so we consider simulations as an alternative approach. We perform an asymptotic analysis. A meaningful comparison of the computational costs needs to take also into account the incertitude of the solution: both the simulations and our method can produce a better result if one is willing to afford a higher number of operations. In order to combine these two aspects in our analysis we consider as metric the product precision times number of operations. Intuitively the larger this product the more expensive is to get a given precision. For the simulations the computational cost is at least proportional to the number of events that are generated, let us denote it by n_E. The incertitude on the final result can be estimated by the amplitude of the confidence interval, that

decreases as $1/\sqrt{n_E}$, then the product precision times number of operations is proportional to $\sqrt{n_E}$ for the simulations. In the case of our approach, the heaviest operation is the solution of the renewal equation. If we adopt the same τ and Δ for all the integrals, we need to calculate the value of the CDF of the miss rate ($G(t)$) in $n_P = \tau/\Delta$ points and then we need to calculate n_P integrals. The integration interval is at most equal to the TTL duration T (see Eq. 13), then each integral requires a number of operations proportional to $n'_P = T/\Delta$. If the value of τ is selected proportionally to T, then the cost of our method is proportional to n_P^2. A naive implementation of the integral as a sum of the function values leads to an error proportional to the amplitude of the time step and then inversely proportional to n'_P or n_P. In conclusion the product precision times the number of operations is proportional to n_P. Then, for a given precision, our method would require a number of points much larger than the number of events to be considered in the corresponding simulation (at least asymptotically). The comparison would then lead to prefer the simulations at least when small incertitude is required (then large n_E and n_P). In reality integrals can be calculated in more sophisticated ways, for example if we adopt Romberg's method, with a slightly larger computation cost, we can get a precision proportional to n_P^{-2}. In this case the product precision times number of operations is a constant for our method, that should be preferred.

7 Applications

In this section we first show how our model can predict the location of a content in a general network. Then we use it to tune the TTL values in order to minimize the size of every cache.

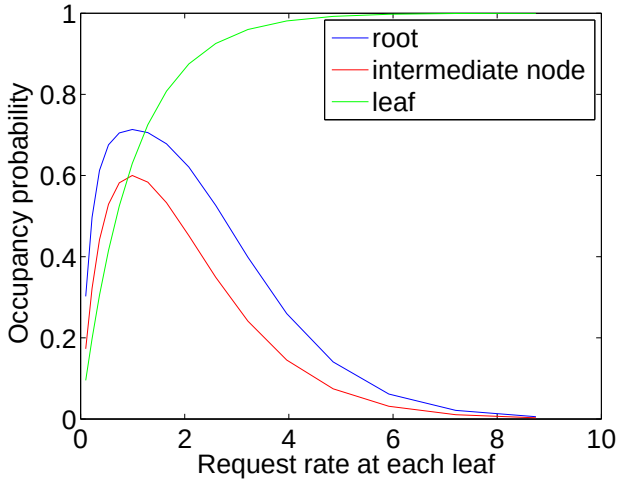


Figure 12: Occupancy versus request rate

We first consider a binary tree with 7 caches: one root, 2 intermediate caches and 4 leaves. Requests arrive only at the leaves according to iid Poisson processes. The timer is deterministic and equal to 1 at all the caches. This topology does not belong to class $\mathcal{N}$, but still Eqns (31)-(32) provide a recursive procedure for calculating numerically all the quantities of interest. In particular, Fig. 12 shows the cache occupancy at the different levels of the tree for different request arrival rates at the leaves.

When the request rate at each leaf is small, requests are very unlikely to be satisfied at the leaves and the content is more likely to be stored at higher-level caches inside the network (closer to the server), that receive a higher aggregate request rate. As the request rate increases, the hit rate at the leaves increase and the miss rate ($m_R = \lambda - h_R$) first increases and then decreases as the hit rate increase becomes the dominant effect. As the rate of forwarded requests starts decreasing, the occupancy at higher level caches decreases. This confirms the intuition that in a cache network popular contents are located closer to the user. At the same time, even in such a simple network, there is no monotonic relation among occupancies at different levels: for example for small request rates the content is more located inside the network (the occupancy is increasing moving from the leaves to the root), but for large request rate, the content is less present at intermediate caches than at the root or at the leaves.

If multiple contents are present in the network, we denote the steady-state probability that content k is present at cache n as $\pi_{n,k}$. The average number of contents at cache n can be calculated simply as $\sum_k \pi_{n,k}$. We call it the total occupancy of cache n and we denote it by q_n. So when request and TTL rates are known, our model can be used to calculate the total cache occupancy and then to size the buffer at each node. Conversely, we can set the TTL values at each cache in order to optimize some performance metric. For example the provider may want to set the TTLs in order to minimize the maximum average cache occupancy. We show how our analysis can be used to solve this problem in a class $\mathcal{N}$ network.

Let us consider a network with N caches and K documents. We denote $\Lambda_{n,k}$ and $H^*_{n,k}(s)$, respectively, the total arrival rate and the LST of the CDF of the overall arrival process for content k at cache n. Then it follows that the total occupancy at cache n is

$$q_n = \sum_{k=1}^{K} \pi^A_{n,k} = \sum_{k=1}^{K} \Lambda_{n,k} \frac{1 - H^*_{n,k}(\mu_n)}{\mu_n}.$$

Our goal is to solve the following optimization problem:

$$\min_{\mu_1,\mu_2,\ldots,\mu_N} \max\{q_n | n = 1, 2, \ldots N\} \text{ s.t. } \sum_{n=1}^{N} q_n = Q, \tag{42}$$

where Q is a constraint on the total (expected) occupancy in the network (the total buffer usage in the network).

It is easy to check that under the optimal setting, each cache has the same total occupancy equals to Q/N. We can then apply our recursive procedure starting from the nodes farthest from the server and determine for each of them μ_n such that

$$q_n = \sum_{k=1}^{K} \Lambda_{n,k} \frac{1 - H^*_{n,k}(\mu_n)}{\mu_n} = \frac{Q}{N}.$$

This equation can be solved numerically by using the Newton method.

As a numerical example we have considered the line-of-simple-trees network of Fig. 7 with $K = 100$ and iid request processes at the leaves. The rate of each request process for a given content has been drawn uniformly at random in the interval $[0.01, 10]$. Fig. 13 shows the selected TTL rates for the nodes 1 and 2 and one leaf (all of them have the same TTL rates because their request arrival processes are identical) for different values of the total occupancy in the network $Q \in \{2, 4, 6, 7.2\}$.

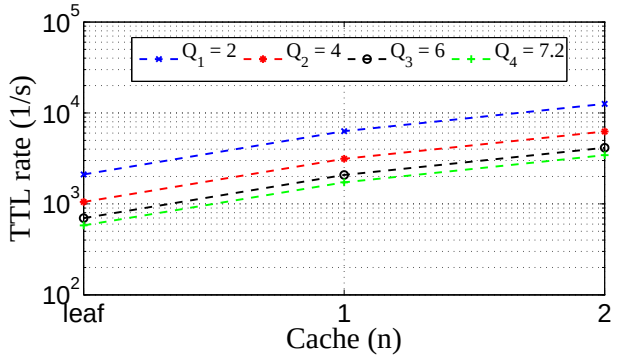


Figure 13: CCN dimensioning: Optimal TTL rates at different caches for different total occupancy in network.

8 Conclusion

In this paper we have developed a set of building blocks for the performance evaluation of hierarchical TTL cache networks where TTLs are set with every request. For some topologies our results are exact but when they are not, the relative errors are extremely small. Thus our approach is promising and we believe capable of accurately modeling a richer class of network topologies. Moreover, although the approach applies to single content caches, we have also demonstrated that it can be used to optimize a multi-content cache network.

Using (25) in (24) gives

$$\begin{aligned} H_{N+1}(t) &= 1-(-1)^N \frac{e^{-\lambda_{N+1}t}}{\Lambda_{N+1}} \prod_{i=1}^{N} \frac{\lambda_i^2}{\mu_i^2-\lambda_i^2} \Bigg[\lambda_{N+1} \sum_{\substack{i_l \in \{0,1\} \\ l=1,\ldots,N}} \prod_{k=1}^{N} (-1)^{i_k} \left(\frac{\mu_k}{\lambda_k}\right)^{2i_k} \\ &\quad \times e^{-(\sum_{k=1}^{N} (\lambda_k^{i_k} + \mu_k^{1-i_k}))t} \\ &\quad - \sum_{i=1}^{N} \frac{\mu_i}{\lambda_i} \sum_{\substack{i_l \in \{0,1\} \\ l=1,\ldots,N, l\neq i}} \prod_{\substack{k=1 \\ k\neq i}}^{N} (-1)^{i_k} \left(\frac{\mu_k}{\lambda_k}\right)^{2i_k} \\ &\quad \times \left(\mu_i e^{-(\lambda_i + \sum_{\substack{k=1 \\ k\neq i}}^{N} (\lambda_k^{i_k} + \mu_k^{1-i_k}))t} - \lambda_i e^{-(\mu_i + \sum_{\substack{k=1 \\ k\neq i}}^{N} (\lambda_k^{i_k} + \mu_k^{1-i_k}))t} \right) \Bigg]. \end{aligned} \tag{43}$$

References

[1] J. R. Bitner, "Heuristics that monotonically organize data structures", *SIAM J. Computing*, **8**, pp. 82-110, 1979.

[2] P. J. Burville and J. F. C. Kingman, "On a model for storage and search", *Journal of Applied Probability*, **10**, pp. 697-701, 1973.

[3] G. Carofiglio, M. Gallo, L. Muscariello and D. Perino, "Modeling data transfer in content-centric networking", *Proc ITC23*, San Francisco, CA, USA, Sep. 6-8, 2011.

[4] H. Che, Y. Tung and Z. Wang, "Hierarchical Web caching systems: modeling, design and experimental results", *IEEE J. on Selected Areas in Communications*, Vol. 20, No. 7, pp. 1305-1314, Sep. 2002.

[5] E. G. Coffman Jr. and P. Jelenkovic, "Performance of the Move-to-Front algorithm with Markov-modulated request sequences", *Operations Research Letters*, **25**, pp. 109-118, 1999.

[6] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes", *Proc. ACM Sigmetrics 1990*, Boulder, CO, USA, May 22-25, 1990, pp. 143-152.

[7] R. P. Dobrow and J. A. Fill, "The move-to-front rule for self-organizing lists with Markov dependent requests", *Discrete Probability and Algorithms*, IMA Volumes in Mathematics and its Applications, D. Aldous, P. Diaconis, J. Spencer, and J. M. Steele (Eds), **72**, pp. 57-80, Springer-Verlag, 1995.

[8] P. Flajolet, D. Gardy and L. Thimonier, "Birthday paradox, coupon collectors, caching algorithms and self-organizing search", *Discrete Applied Mathematics*, **39**, pp. 207-229, 1992. First version appeared as an INRIA Tech. Report, No. 720, Aug. 1987.

[9] W. J. Hendricks, "The stationary distribution of an interesting Markov chain", *Journal of Applied Probability*, **9**, pp. 231-233, 1972.

[10] J. A. Fill, "Limits and rate of convergence for the distribution of search cost under the move-to-front rule", *Theoretical Computer Science*, **176**, pp. 185-206, 1996.

[11] P. Jelenkovic, "Asymptotic approximation of the move-to-front search cost distribution and least-recently used caching fault probabilities", *The Annals of Probability*, Vol. 9, No. 2, pp. 430-464, 1999.

[12] P. Jelenkovic and A. Radovanović, "Least-recently used caching with dependent requests", *Theoretical Computer Science*, **326**, pp. 293-327, 2004.

[13] P. Jelenkovic and A. Radovanović and M. Squillante, "Critical sizing of LRU caches with dependent requests", *Journal of Applied Probability*, Vol. 43, No. 4, pp. 1013-1027, December 2006.

[14] J. Jung, A. W. Berger and H. Balakrishnan, "Modeling TTL-based Internet Caches", *Proc. IEEE Infocom 2003*, San Francisco, CA, USA, Mar. 30 - Apr. 3, 2003.

[15] J. Jung, E. Sit, H. Balakrishnan and R. Morris, "DNS performance and the effectiveness of caching", *Proc. ACM SIGCOMM Workshop on Internet Measurement (IMW '01)*, New York, NY, USA, Nov. 1-2, 2001.

[16] W. F. King, "Analysis of demand paging algorithm", *Information Processing*, Vol. 71, pp. 485-490, 1972. [Appeared first under the same title as IBM Research Report, RC 3288, Mar. 17, 1971.]

[17] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules", *Advances in Applied Mathematics*, Vol. 6, pp.4-22, 1985.

[18] I. Lassoued, A. Krifa, C. Barakat and K. Avrachenkov, "Network-wide monitoring through self-configuring adapative system", *Proc. IEEE Infocom 2011*, Shanghai, China, Apr. 10-15, 2011.

[19] A. T. Lawrance, "Dependency of intervals between events in superposition processes", *Journal of the Royal Statistical Society*, Series B (Methodological), Vol. 35, No. 2, pp. 306-315, 1973.

[20] J. Mc Cabe, "On serial files with relocable records", *Oper. Res.*, **13**, pp. 609-618, 1965.

[21] E. J. Rosensweig, J. Kurose and D. Towsley, "Approximate models for general cache networks", *Proc. IEEE Infocom 2010*, San Diego, CA, USA, Mar. 15-19, 2010.

[22] V. Jacobson, D. K. Smetters, J. D. Thorntorn, M. Plass, N. Briggs and R. L. Braynard, "Networking named content", *Proc. ACM CoNEXT 2009*, Rome, Italy, Dec. 1-4, 2009.

[23] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble and M. Levy, "An analysis of Internet content delivery systems". *SIGOPS Operating System Review*, Vol. 36, isse SI, pp. 315-327, 2002.

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399